CENTRO UNIVERSITÁRIO UNIVATES

CURSO DE ANALISE E DESENVOLVIMENTO DE SISTEMAS

**ESTUDO DE CASO DE UM SISTEMA DE ORGANIZAÇÕES DE AÇÕES COM BASE NA FERRAMENTA 5W2H**

Lucas Leandro de Moura

Lajeado, Junho de 2017

Lucas Leandro de Moura

**ESTUDO DE CASO DE UM SISTEMA DE ORGANIZAÇÕES DE AÇÕES COM BASE NA FERRAMENTA 5W2H**

Trabalho apresentado na disciplina Programação Avançada, no curso de tecnologia em Analise e Desenvolvimento de Sistemas, do Centro Universitário UNIVATES.

Professor: Prof. Ms. Juliano Dertzbacher

Lajeado, Março de 2017

1. **INTRODUÇÃO**

Este documento foi elaborado para uma tarefa do módulo de Programação Avançada do curso de Análise e Desenvolvimento de Sistemas da Univates. Com isso, em um estudo breve sobre uma situação real solicitada, identifiquei a necessidade da elaboração de um plano de um sistema que realiza o controle de ações a serem realizadas pelas corporações utilizando a ferramenta administrativa 5W2H. A ferramenta 5W2H, se posiciona em base de algumas perguntas básicas para a organização das atividades que a corporação deverá adotar, que são: Quem? O que? Onde? Quando? Por que? Como?.

Este método é muito simples, que agiliza todas ações de uma empresa, trazendo uma visão rápida e simples de como as corporações poderão alcançar as suas metas e indicadores de desempenho.

* 1. **Análise de Requisitos**

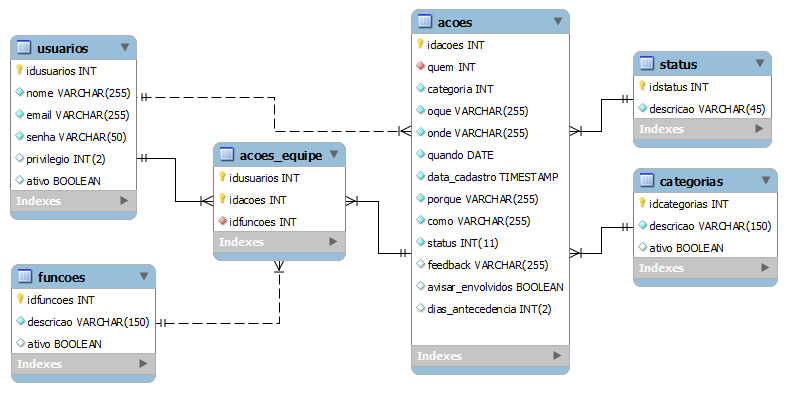
**Objetivo:** Desenvolver um sistema para controlar ações com a ferramenta de gestão estratégica 5W2H.

**Requisitos funcionais**

1. Manter cadastros de *Stakeholders;*
2. Manter cadastro de categorias;
3. Manter cadastro de funções;
4. Gerenciar níveis de permissão dos *Stakeholders*;
5. Gerenciar equipes das ações;
6. Manter o cadastro de ações dentro da grade 5W2H;
7. Permitir gerar relatórios através de filtros: Quem? O que? Onde? Quando? Por que? Como? Concluído?;
8. Avisar os *Stakeholders* de ações que estão atrasadas;
9. Permitir que o *Stakeholder* conclua a ação informando o *feedback* da atividade realizada;

**Requisitos Não-funcionais;**

1. Utilizar linguagem *Java*;
2. Utilizar banco de dados *PostgreSQL*;
3. Gerar relatório através da biblioteca *IReport*;
4. Validar campos de data;
5. Validar como obrigatório os campos quem, o que, onde, quando, por que, como;
6. Validar o acesso dos usuários através de autenticação;
   1. **Modelo Relacional**



* 1. ***Dump SQL***

CREATE TABLE acoes (

quem integer,

categoria integer NOT NULL,

oque character varying(255) NOT NULL,

idacoes integer NOT NULL,

onde character varying(255) NOT NULL,

quando date NOT NULL,

data\_cadastro timestamp(0) without time zone DEFAULT now(),

porque character varying(255) NOT NULL,

como character varying(255) NOT NULL,

status integer DEFAULT 0 NOT NULL,

feedback character varying(255),

avisa\_envolvidos boolean DEFAULT false,

dias\_antecedencia integer

);

ALTER TABLE acoes OWNER TO postgres;

CREATE TABLE acoes\_equipe (

idusuarios integer NOT NULL,

idacoes integer NOT NULL,

idfuncoes integer

);

ALTER TABLE acoes\_equipe OWNER TO postgres;

CREATE TABLE categorias (

idcategorias integer NOT NULL,

descricao character varying(150) NOT NULL,

ativo boolean DEFAULT true

);

ALTER TABLE categorias OWNER TO postgres;

CREATE SEQUENCE categorias\_idcategorias\_seq

START WITH 1

INCREMENT BY 1

NO MINVALUE

NO MAXVALUE

CACHE 1;

ALTER TABLE categorias\_idcategorias\_seq OWNER TO postgres;

ALTER SEQUENCE categorias\_idcategorias\_seq OWNED BY categorias.idcategorias;

CREATE TABLE funcoes (

idfuncoes integer NOT NULL,

descricao character varying(150) NOT NULL,

ativo boolean DEFAULT true

);

ALTER TABLE funcoes OWNER TO postgres;

CREATE SEQUENCE funcoes\_idfuncoes\_seq

START WITH 1

INCREMENT BY 1

NO MINVALUE

NO MAXVALUE

CACHE 1;

ALTER TABLE funcoes\_idfuncoes\_seq OWNER TO postgres;

ALTER SEQUENCE funcoes\_idfuncoes\_seq OWNED BY funcoes.idfuncoes;

CREATE TABLE status (

idstatus integer NOT NULL,

descricao character varying(45) NOT NULL

);

ALTER TABLE status OWNER TO postgres;

CREATE SEQUENCE status\_idstatus\_seq

START WITH 1

INCREMENT BY 1

NO MINVALUE

NO MAXVALUE

CACHE 1;

ALTER TABLE status\_idstatus\_seq OWNER TO postgres;

ALTER SEQUENCE status\_idstatus\_seq OWNED BY status.idstatus;

CREATE SEQUENCE table\_idacoes\_seq

START WITH 1

INCREMENT BY 1

NO MINVALUE

NO MAXVALUE

CACHE 1;

ALTER TABLE table\_idacoes\_seq OWNER TO postgres;

ALTER SEQUENCE table\_idacoes\_seq OWNED BY acoes.idacoes;

CREATE TABLE usuarios (

idusuarios integer NOT NULL,

nome character varying(255) NOT NULL,

email character varying(255) NOT NULL,

senha character varying(50) NOT NULL,

privilegio integer DEFAULT 0 NOT NULL,

ativo boolean DEFAULT true

);

ALTER TABLE usuarios OWNER TO postgres;

CREATE SEQUENCE usuarios\_idusuarios\_seq

START WITH 1

INCREMENT BY 1

NO MINVALUE

NO MAXVALUE

CACHE 1;

ALTER TABLE usuarios\_idusuarios\_seq OWNER TO postgres;

ALTER SEQUENCE usuarios\_idusuarios\_seq OWNED BY usuarios.idusuarios;

ALTER TABLE ONLY acoes ALTER COLUMN idacoes SET DEFAULT nextval('table\_idacoes\_seq'::regclass);

ALTER TABLE ONLY categorias ALTER COLUMN idcategorias SET DEFAULT nextval('categorias\_idcategorias\_seq'::regclass);

ALTER TABLE ONLY funcoes ALTER COLUMN idfuncoes SET DEFAULT nextval('funcoes\_idfuncoes\_seq'::regclass);

ALTER TABLE ONLY status ALTER COLUMN idstatus SET DEFAULT nextval('status\_idstatus\_seq'::regclass);

ALTER TABLE ONLY usuarios ALTER COLUMN idusuarios SET DEFAULT nextval('usuarios\_idusuarios\_seq'::regclass);

ALTER TABLE ONLY acoes\_equipe

ADD CONSTRAINT acoes\_equipe\_pkey PRIMARY KEY (idusuarios, idacoes);

ALTER TABLE ONLY categorias

ADD CONSTRAINT categorias\_pkey PRIMARY KEY (idcategorias);

ALTER TABLE ONLY funcoes

ADD CONSTRAINT funcoes\_pkey PRIMARY KEY (idfuncoes);

ALTER TABLE ONLY status

ADD CONSTRAINT status\_pkey PRIMARY KEY (idstatus);

ALTER TABLE ONLY acoes

ADD CONSTRAINT table\_pkey PRIMARY KEY (idacoes);

ALTER TABLE ONLY usuarios

ADD CONSTRAINT usuarios\_pkey PRIMARY KEY (idusuarios);

ALTER TABLE ONLY acoes\_equipe

ADD CONSTRAINT acoes\_equipe\_fk FOREIGN KEY (idacoes) REFERENCES acoes(idacoes);

ALTER TABLE ONLY acoes\_equipe

ADD CONSTRAINT acoes\_equipe\_fk1 FOREIGN KEY (idfuncoes) REFERENCES funcoes(idfuncoes);

ALTER TABLE ONLY acoes

ADD CONSTRAINT acoes\_fk FOREIGN KEY (status) REFERENCES status(idstatus);

ALTER TABLE ONLY acoes

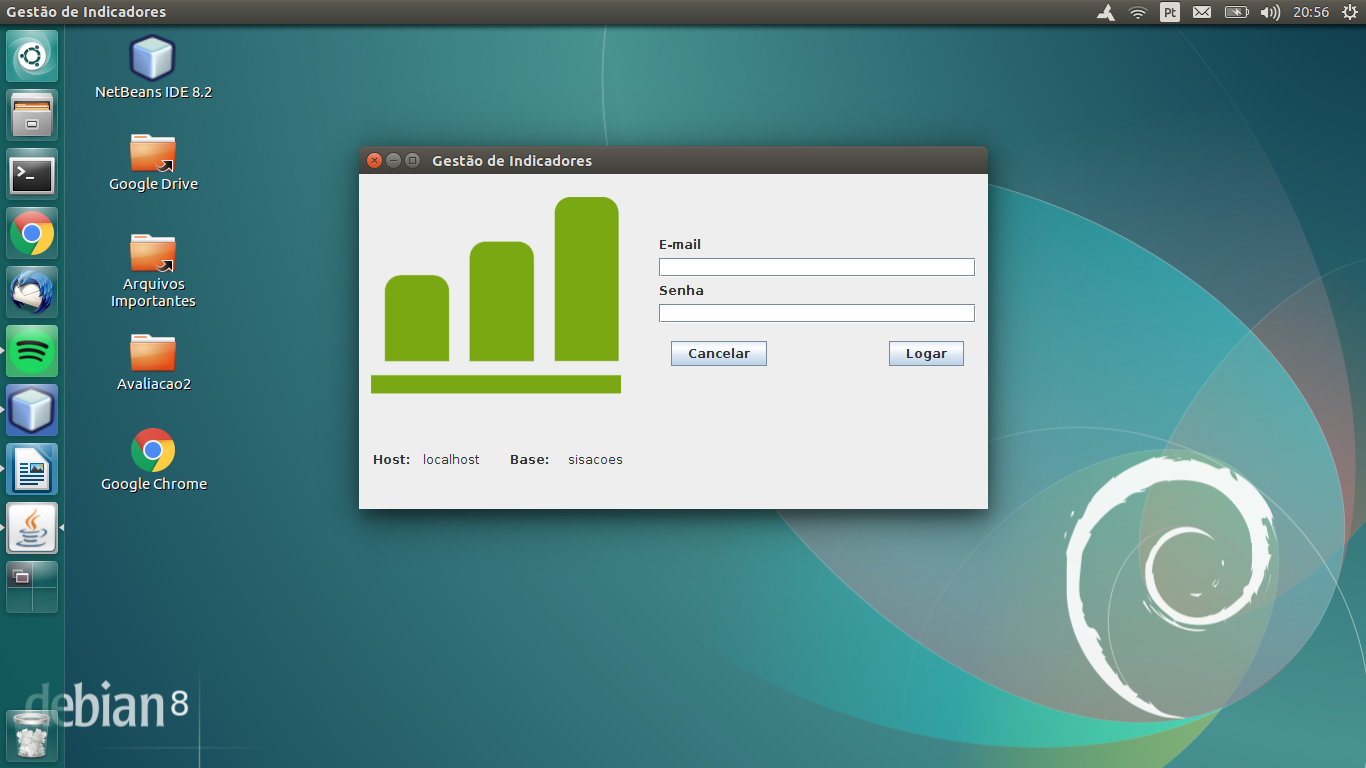
ADD CONSTRAINT acoes\_fk1 FOREIGN KEY (categoria) REFERENCES categorias(idcategorias);

ALTER TABLE ONLY acoes

ADD CONSTRAINT acoes\_fk2 FOREIGN KEY (quem) REFERENCES usuarios(idusuarios);

**4. Interfaces**

a) Tela de configuração de acesso ao banco de dados, instalado na máquina.

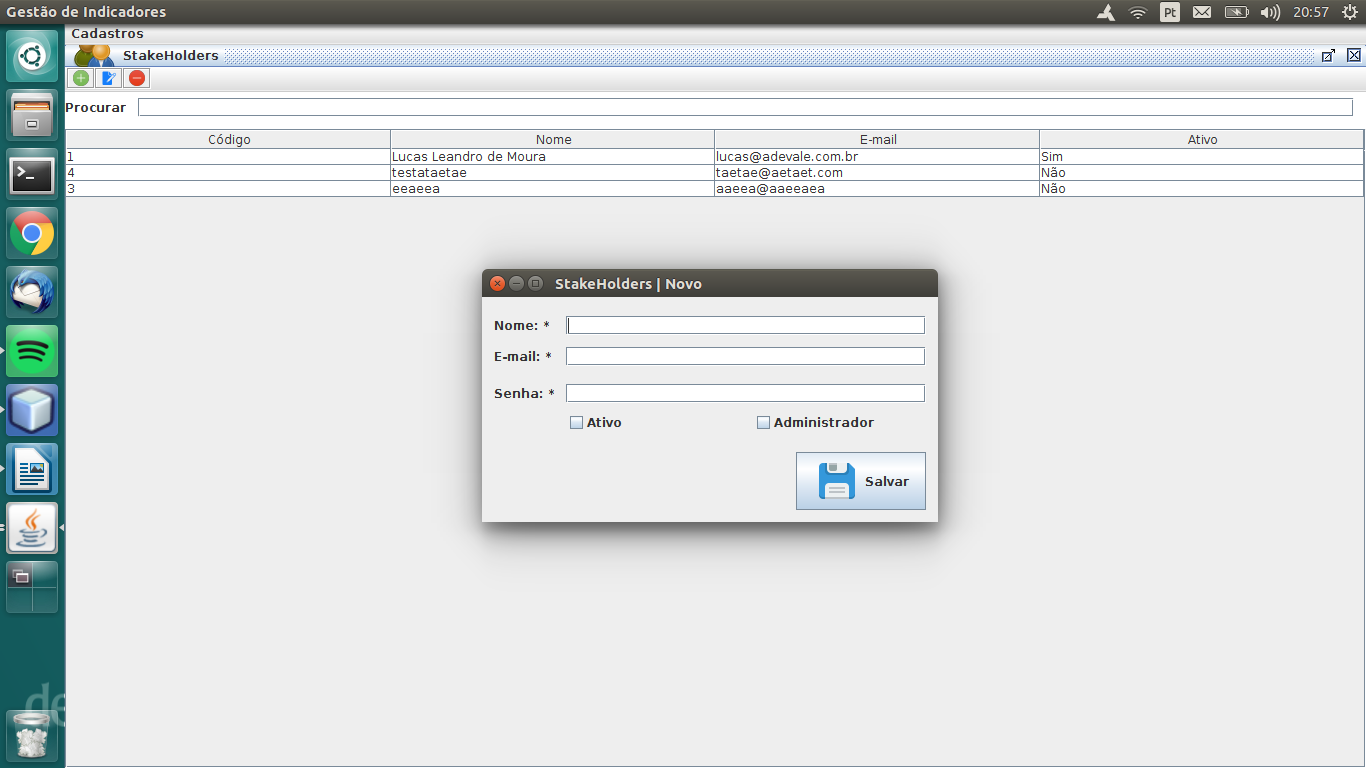


b) Antes de entrar na tela principal, o sistema solicita para que o usuário informe o seu login e senha. De padrão foi inserido o usuário [admin@admin.com.br](mailto:admin@admin.com.br) e senha 1234.

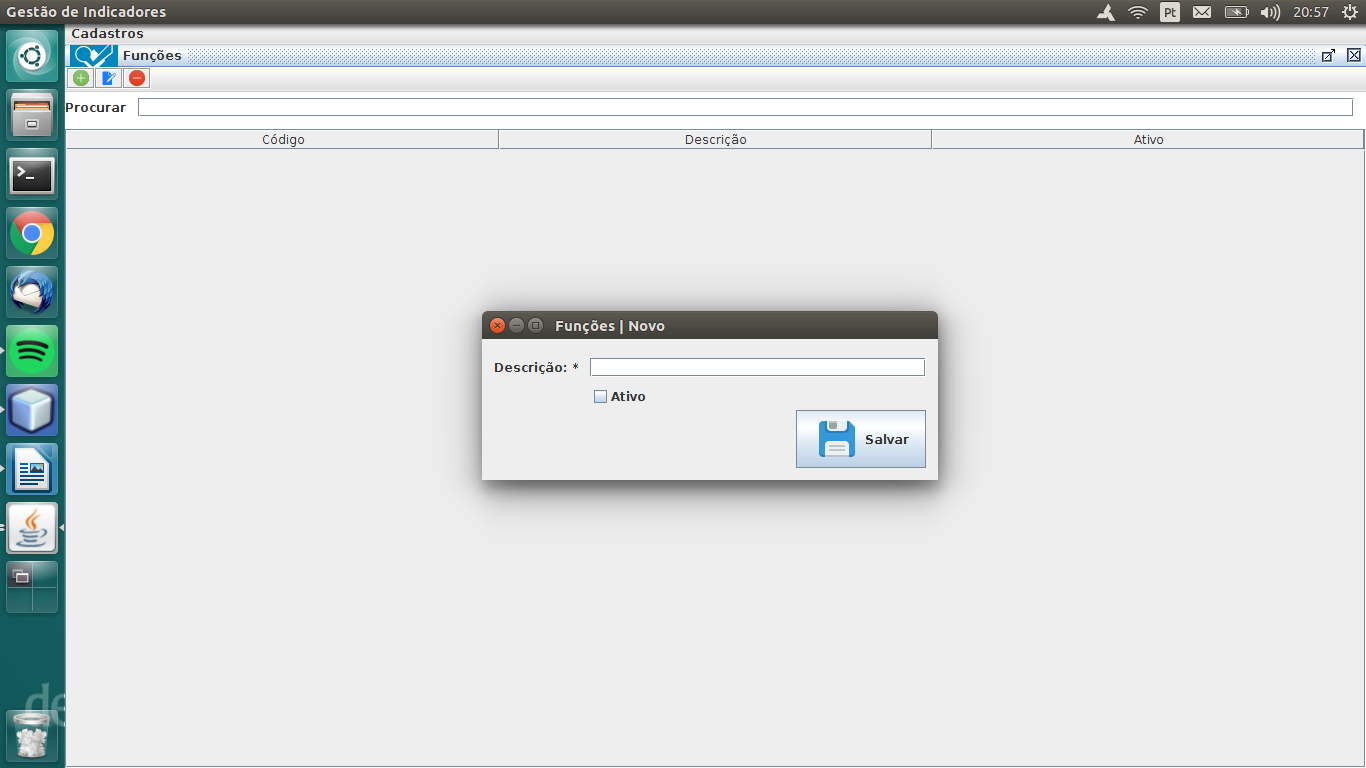


c) Cadastro de stakeholders

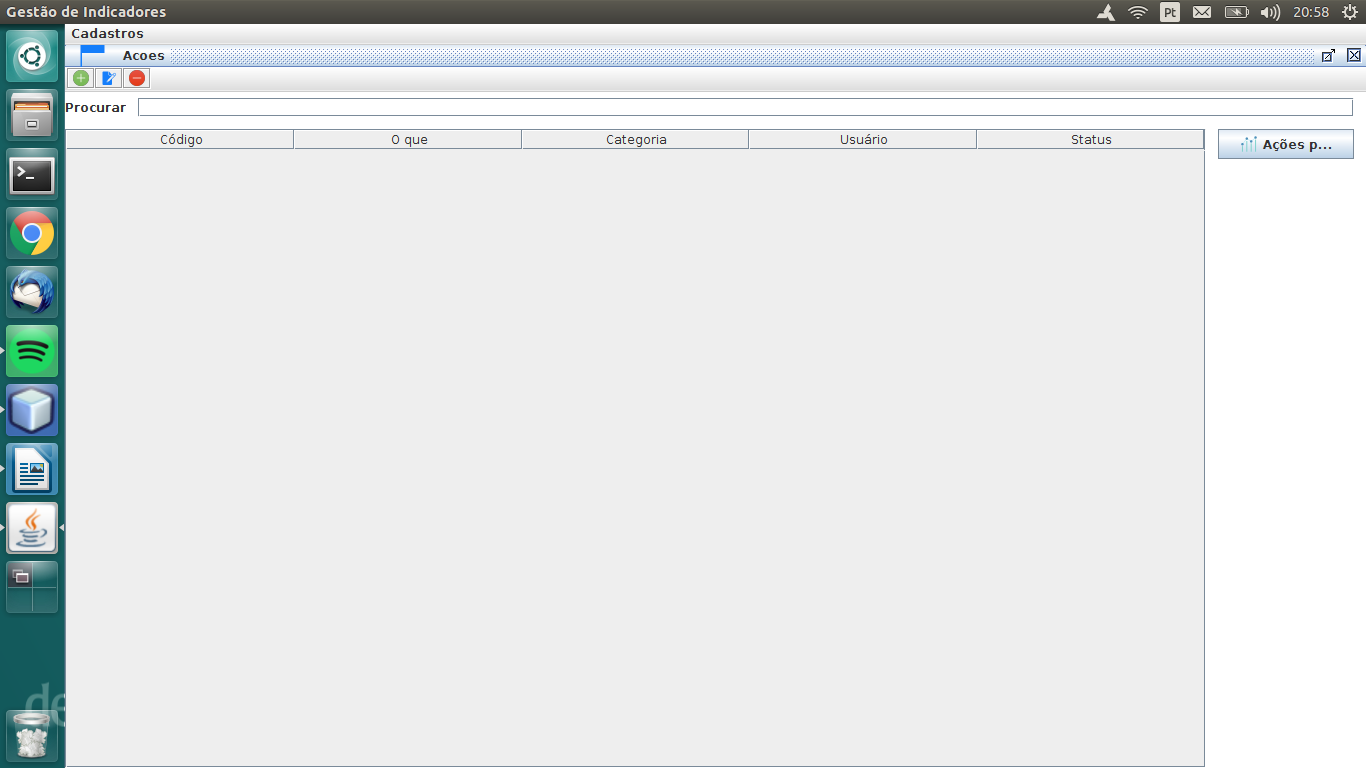
Todos os campos obrigatórios, são demarcados pelo \*.



d) Cadastro de funções, é responsável por cadastrar funções que serão utilizadas na classificação dos participantes dentro daquele evento.



e) No cadastro de ações, no qual é a tela que é a parte central do sistema. O mesmo poderá manter os registros das atividades, que terão que ser tomadas e controladas.



**10. Referencial Bibliográfico**

5W2h Planejamento Estratégico, Disponível em: <https://sites.google.com/site/planejaweb/5w2h>. Acessado em: 07 Mar. 2017.

Métodos e técnicas de planejamento:

<http://www.valentim.pro.br/data/documents/Metodos\_Tecnicas\_Planejamento\_1.pdf>. Acessado em: 07 Mar. 2017.